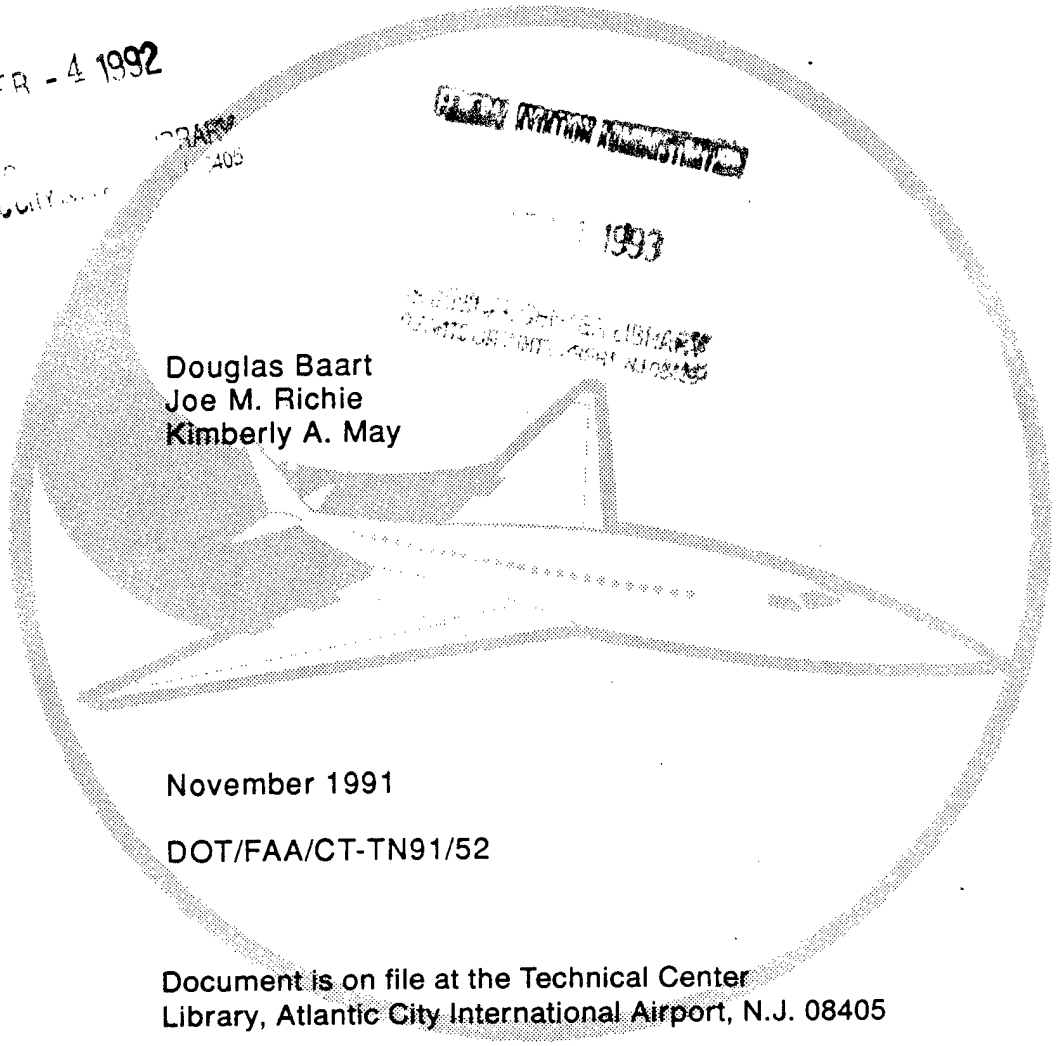
Cost of Delay Module



Douglas Baart
Joe M. Richie
Kimberly A. May

November 1991

DOT/FAA/CT-TN91/52

U.S. Department of Transportation
Federal Aviation Administration

Technical Center
Atlantic City International Airport, N.J. 08405

NOTICE

This document is disseminated under the sponsorship of the U.S. Department of Transportation in the interest of information exchange. The United States Government assumes no liability for the contents or use thereof.

The United States Government does not endorse products or manufacturers. Trade or manufacturers' names appear herein solely because they are considered essential to the objective of this report.

TABLE OF CONTENTS

EXECUTIVE SUMMARY

This paper addresses the cost of the delay module that was developed by Federal Aviation Administration Technical Center to be incorporated into the National Airspace System Analysis Capability (NASPAC) model. This module was developed to address the savings which could be realized when changes are made to the Air Traffic Control (ATC) System. The purpose of this module is to translate delay into a cost metric and, thus, give policy makers a better understanding of potential cost saving measures. These cost saving measures are a direct result of an operational or procedural change to the ATC System. Cost estimates for major air carriers using the National Airspace System (NAS) were derived from Form 41 (operating expenses) data provided by the Office of Airline Statistics. General aviation and military cost estimates were derived from the Economic Values for Evaluation of Federal Aviation Administration Investment and Regulatory Programs, FAA-APO-89-10. The cost of the delay module has been tested and is fully operational with the latest release of NASPAC.

1. INTRODUCTION.

1.1 BACKGROUND.

The MITRE Corporation, at the directional guidance of the Federal Aviation Administration (FAA), has developed an analysis tool called National Airspace Systems Performance Analysis Capability (NASPAC) Simulation Modeling System. NASPAC is a discrete-event simulation model which captures flight operation events occurring throughout the National Airspace System (NAS) for an entire day. NASPAC was designed to study system-wide performance of the NAS under existing operational conditions and to predict its performance under a proposed set of changes to the NAS Air Traffic Control (ATC) System. Alternate operating policies can be evaluated to determine which best meet the needs of the ATC System in terms of system performance. The model is used to study the relationships and interactions that occur between components of the ATC System. An evaluation of proposed system changes are necessary before any procedural or operational changes to the ATC System are implemented.

The NASPAC simulation model is used to study the NAS on a macro level and the effects of projected changes to it in demand, capacity, and operational procedures. The model describes NAS performance in terms of throughput and delay at modeled entities. Delay occurs whenever aircraft must wait to use a resource in the ATC System (e.g., arrival runway, departure runway, arrival departure fix, sectors, etc.) as demand for that resource exceeds its capacity to serve aircraft. There are two types of delays modeled by NASPAC. Technical delay is the delay absorbed by aircraft while waiting to use ATC resources. Technical delay can be encountered at airports, fixes, and in sectors and is the type of delay that FAA regulations and procedures may affect. Effective delay measures the difference between planned (scheduled) and actual arrival time. Effective arrival delay is the type of delay that directly affects passengers.

NASPAC measures throughput and delay as a quantitative basis for decision-making concerning changes to the ATC System. Although these metrics provide an analyst with estimates of system performance, the amount of money saved along with safety considerations generally determine whether such changes are implemented. Delays reported in the NASPAC simulation can be translated into a cost metric. These estimates are designed to evaluate monetary gains as a result of reducing delay. The value of time saved by reducing delays is important in determining whether investment and regulatory decisions affecting the aviation system are economically rational.

NASPAC can be applied to a wide variety of long-term strategic planning studies for the NAS. Each application must be customized to reflect specific issues of the study. The model is customized through system parameter definitions and scenario development. Each study will include the following two generalized scenarios:

Baseline scenario: Provides baseline throughput and delays for the existing system.

Alternative scenario: Provides throughput and delay for the NAS for the proposed alternatives under study.

The cost of the delay module will provide the analyst with estimates of delay costs when comparisons are made between baseline conditions and proposed changes to the ATC System. When the cost of delay module is executed for a study, delay cost estimates for each scenario developed will be provided. These estimates include cost of technical delay and cost of effective delay. These estimates can be compared between baseline and alternate proposals to see if any benefits will be realized.

This document addresses the procedure used in determining cost estimates for each type of delay that the NASPAC simulation model produces.

1.2 PURPOSE.

A milestone identified in the NASPAC transition plan is to complete and integrate a cost of delay module into the model with Release 2. This module provides cost/savings estimates for both the technical and effective delay components modeled in the system. It translates each type of delay recorded by the model into a cost figure.

This effort can be summarized into three main activities. These activities are not trivial and each contain a number of tasks.

a. Develop an estimate of cost per time unit of delay based on operational expenses comprised of airborne and ground operations and on passenger time.

b. Develop an algorithm that applies the cost estimate to the delays accrued at each ATC resource modeled.

c. Summarize the cost metric in a usable format.

2. COST ARRAY DEVELOPMENT.

When estimating technical unit/cost values, care must be given to include only operational costs. Delay affecting passengers will be addressed in the effective cost estimate. The total delay savings associated with a study will be the sum of the technical delay and passenger delay savings.

2.1 PASSENGER DELAY.

Effective delay measures the difference between Official Airline Guide scheduled arrival times and the arrival times measured by the simulation model. This metric measures a cost to passengers as a result of lost time. The savings in cost associated with a study would

be the time savings to passengers realized from the reduction in effective delay. The cost of effective delay is a constant derived from the Office of Aviation Policy and Plans (APO-1), Economic Analysis Branch (APO-220). The constant ($39.50/hour) used in the cost of delay module measures the cost per hour of a delay incurred by a passenger. Passenger utilization factors derived from report FAA-APO-89-10 (Economic Values For Evaluation of Federal Aviation Administration Investment and Regulatory Programs) were used to determine the average number of passengers aboard a flight.

2.2 AIRBORNE DELAY.

Airborne and ground delay costs were calculated for air carriers from Form 41 - **Traffic and Capacity Data Summarized by Aircraft Type** (schedule T2) and **Aircraft Operating Expenses** (schedule P05) from the Office of Airline Statistics. These reports are accumulated on a quarterly basis for most of the major air carriers which use the NAS. Total operating expenses for each carrier and aircraft reported were used in developing the cost arrays. Total flying hours aggregated by air carrier and aircraft were determined from schedule T-2 and used in conjunction with the operating expense data (schedule P-05) to produce costs estimates.

2.3 GROUND DELAY.

Ground operating costs were derived by first determining the time each aircraft spent on the ground. This was determined by taking the difference between the gate to gate hours as recorded in schedule T-2 and the total flying hours. Flying operating expenses were subtracted from total operating expense to produce estimates of ground expense. These values were used with the time an aircraft spends on the ground to produce a unit cost estimate for all ground operations.

General aviation and military cost estimates for airborne and ground operations were derived from the FAA-APO-89-10 document "Economic Values For Evaluation of Federal Aviation Administration Investment and Regulatory Programs."

Form 41 data contain the following metrics:

Aircraft Operating Expenses -- Flying
- Pilots and copilots
- Other flight personnel
- Flight attendance expense (passenger service expense)
- Personnel expense
- Aircraft fuels
- Aircraft oils
- Employee benefits and pensions
- Taxes -- payroll
- Taxes -- other than payroll

Total Direct Maintenance -- Flight Equipment
Available Seat-Miles
Revenue Aircraft Miles Flown
Total Aircraft Hours (Airborne)
Aircraft Hours (Ramp-to-Ramp)
Aircraft Fuels Issued (Gallons)

3. COST OF DELAY MODULE DEVELOPMENT.

3.1 MODULE INPUT.

Delay metrics measured at ATC resources are recorded from the NASPAC model and provide the delay information needed by the cost of the delay module. This includes whether the delay was a technical delay (airborne or ground) or an effective delay type. When delay occurs, the following information is recorded to a file:

a. Simulation clock time (time of simulation day)
b. ATC resource type (determines delay type)
c. Resource name
d. Aircraft tail number
e. Minutes of delay

Example Delay File

a	b	c	d	e
1490.34	AED	BOS	3729-10	0.336
1490.34	APD	DEN	1498- 6	1.223
1490.36	AED	ORD	1662- 9	3.226
1490.37	DFX	ORD	664-10	2.654
1490.37	RST	ATL	1233-10	3.221

There are two types of delay modeled by NASPAC: technical delay and effective delay. Effective arrival delay is recorded for each airport. Technical delay can be subdivided into airborne and ground delay. Delay performance data are recorded for the following resources:

Technical Delay

Airborne Delay:
Departure fix delay for each departure fix
Restriction delay for each restriction boundary
Sector entry delay for each sector
Arrival fix delay for each arrival fix
Airport arrival delay for each airport

Ground Delay:
Airport departure delay for each airport

Effective Delay:
Effective arrival delay for each airport
Effective departure delay for each airport

In addition to the delay file, there is a cost array file consisting of airborne and ground costs for selected air carriers, general aviation, and military operations. The cost array file contain operating costs per minute ($/min) for each carrier and aircraft.

Example Cost Array File

```
115.33  COA  DC9
 78.65  COA  B727
145.33  COA  B747
132.45  COA  A300
112.44  COA  DC10
```

The third input file is the equipment category file. This file consist of tail numbers and the air carrier and aircraft associated with those tail numbers. The tail numbers are generated by the NASPAC model for each run and are used by the cost of delay module to determine the air carrier and aircraft type.

Example Equipment Category File

```
1233 COA B747
1766 AAL B727
3456 PAA DC10
1022 AAL B727
1900 COA DC10
```

3.2 DATA PREPARATION.

The cost of delay module stores the cost figures into arrays for each of the airborne and ground delay types. This is done for each carrier and aircraft in which cost information is available. The cost estimates are accessed by addressing the appropriate carrier and/or aircraft index of the cost arrays. The equipment category file is also stored in an array and is used to translate a tail number into an air carrier and aircraft type. Each record from the delay file that is produced by the NASPAC model is processed sequentially as it appears in the file.

3.3 DATA PROCESSING.

The first procedure is to identify the air carrier and aircraft from the tail number. This is done by referencing the indexed tail number to the equipment category array. After the carrier and airframe are determined, the type of delay is categorized according to airborne, ground, or effective delay. Departure fix, restriction, sector entry, arrival fix, and airport arrival delays are considered airborne technical delays; airport departure delay is considered a ground technical delay. Effective delays include airport arrivals and airport departures.

Airborne and ground delays are translated into a cost figure by referencing the appropriate cost array (airborne,ground) based on carrier and aircraft type and multiplying it by its corresponding delay. If no cost information is present for that carrier/aircraft combination, then the cost of that delay is determined by the aircraft alone. If no cost information exist on that aircraft then it is recorded to a "no match.ext" file and the record is excluded from the analysis. Since the baseline and proposal scenario run from the same aircraft file for a given demand profile, the module will exclude the same records from each scenario and, thus, comparisons can be made. The "noname.ext" file is used to identify the aircraft types which contained no available cost information.

A constant of $39.50/hour was used to evaluate the cost of each passenger delay (effective delay). Load factors were used to estimate the number of passengers being served for each aircraft type. Passenger delay cost estimates were calculated by multiplying the average number of passengers on each flight, the total effective delay, and $39.50/hour.

3.4 MODULE OUTPUT.

A summary file is developed after all delay costs have been accumulated for each record in the delay file. Total delay time in minutes and cost of those delays for each type of delay is summarized and aggregated by air carrier, general aviation, and military operations. Those aircraft in which no cost information were available are tabulated to determine how much of the delay is translated into cost measures. These categories are broken down by percentages as a means of assessing missing information.

Example Summary File

**********Effective Delay**********

Effective Arrival Delay	General Aviation	24843 min cost $	86355
	Military	66 min cost $	344
	Air Carrier	172009 min cost $	944906
	Totals	196918 min cost $	1031605

**********Technical Delay Airborne**********

Departure Fix Delay	General Aviation	26 min cost $	386
	Military	3 min cost $	22
	Air Carrier	971 min cost $	44059

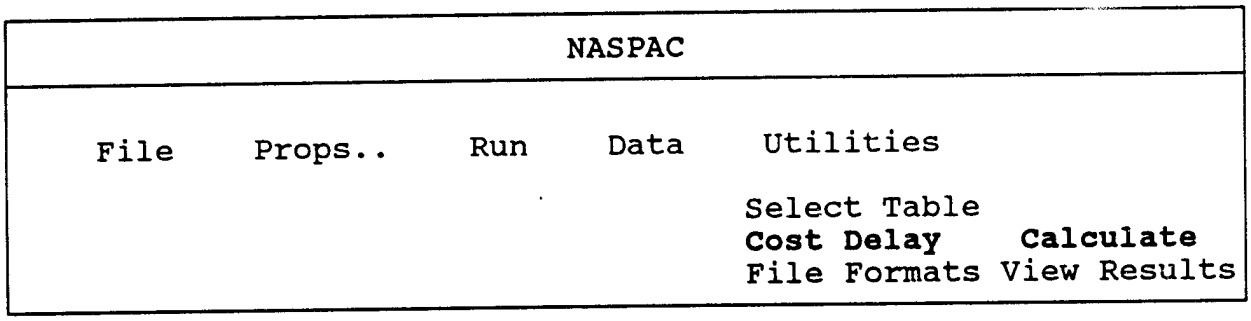
4. VALIDITY ISSUES.

Higher economic costs occur when larger aircraft must absorb delay at a particular ATC resource. Aircraft type, therefore, becomes an important factor in calculating operational costs and in determining loading factors essential for calculating passenger costs. There has been some concern over the validity of using delay information from the NASPAC model at this level of detail for determining costs. The NASPAC simulation model has been validated on a macro-level only. NASPAC builds aircraft itineraries by linking flights based on OAG data, which includes aircraft type information. The validation effort consists of comparing aircraft type sequencing at a resource to independent data sources such as: Host Z sequencing, terminal area sequencing data, and priority landing schemes to identify any significant differences. There is expected to be localized differences, but no significant differences are expected on the aggregate. The cost estimates of delay should be considered as "ballpark" estimates since they were derived from a macro model (NASPAC).

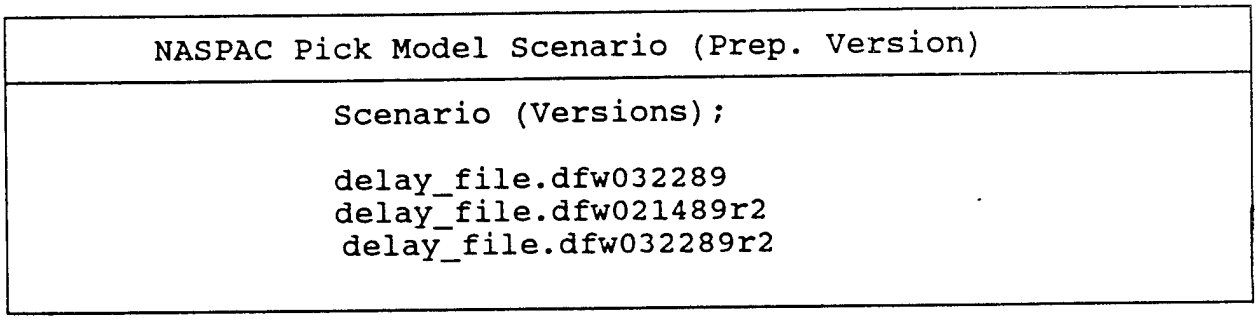
An additional area of concern is the convergence of the model's stochastic components. Analysis of the variability in the output from the NASPAC simulation model has indicated that multiple simulation runs are needed to address statistical reliability issues. The effect of output variability on the implementation of the cost of delay module needs to be addressed. At present, FY-92 plans call for development and implementation of a post processing module which summarizes results and provides statistical measures of precision.

5. USER INTERFACE IN RELEASE 2.

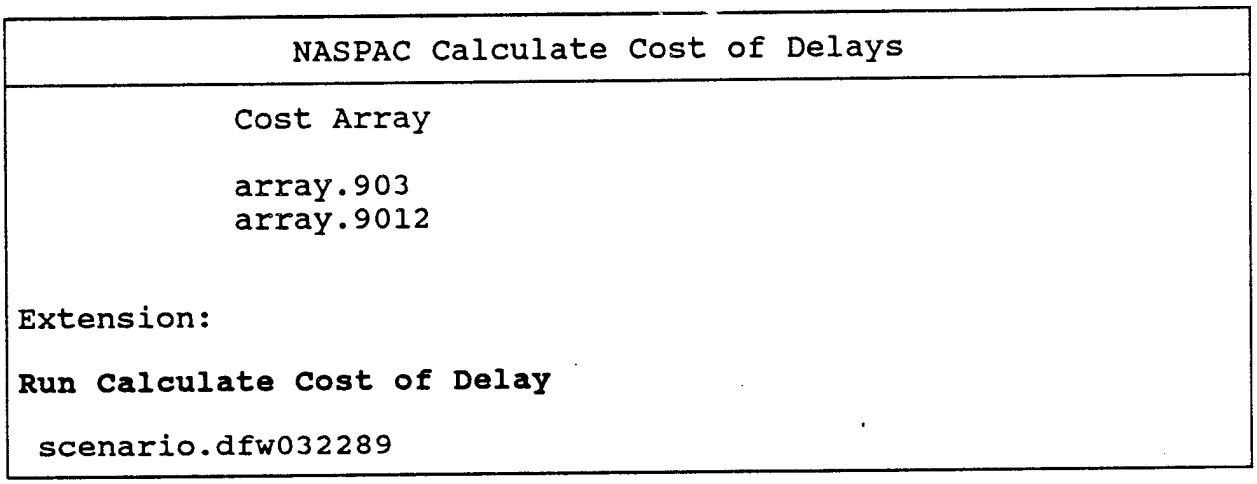
If the user wishes to determine the cost of delay for a given scenario the utilities heading of the main menu should be selected. The **Cost Delay** should then be selected along with **Calculate**.



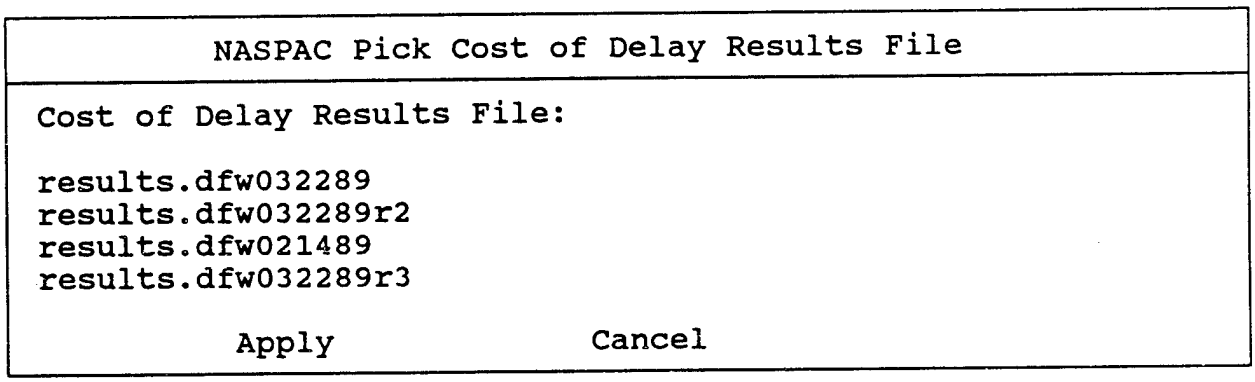
A scenario file must be defined in order to determine which simulation the cost estimates will be applied to.



The user may then select from several cost array files in which different quarterly statistics by year and month are provided. The user will be asked to provide a file name extension in which the output results file may be referenced by. After the file name extension is given, the Run Calculate Cost of Delay should be selected.



A running cost module will appear at the user prompt and a finished running cost module will appear when the module finishes. At this point the output results may be viewed by selecting the appropriate output file. This is accomplished by selecting the review results from the Main Menu. In addition to evaluating the results file, the nomatch file can be examined. This file contains all the aircraft with no available cost information.



6. FUTURE DEVELOPMENT EFFORTS.

While comparisons of cost of delay summary files produce reasonable daily estimates of cost savings, the need to annualize these figures must be addressed. However, the evaluation of cost saving measures for the entire year is not a trivial issue. The cost associated with fuel, taxes, crew salaries, and maintenance are subject to change throughout the year. In addition, each scenario run of the NASPAC model generate specific conditions based on the assumptions used in that scenario which may not be representative of the entire year. The future cost of the delay module should be flexible in order to quantify cost figures for a given year that reflect changes to the economy.

7. CONCLUSIONS.

The overall objective is to develop a cost of the delay module that can be applied in the post-processing stage of the data analysis. The cost of the delay module will be activated upon the users request within the menu driven system of Release 2. The unit cost of operation for any carrier/aircraft configuration is sensitive to the condition of the economy. Factors such as fuel, salary, maintenance, depreciation, etc., will be affected. As a result, NASPAC customers can select from a number of cost arrays which reflect different quarterly cost statistics. Cost analysis as performed by the cost module is not intended to be a cost-benefit analysis, but a measure of the loss/savings that result on an operational level associated with a change to the ATC System. NASPAC is designed to evaluate the difference between the way the ATC System operates currently and how it will operate with a proposed ATC change. Absolute cost saving measures can be determined by comparing total cost figures for each scenario tested.

APPENDIX A

SOURCE LISTING COST OF DELAY MODULE

```
program cost_mod; { program translates delay into cost metric }
                  { airborne and ground delays are transistioned }
                  { Form 41 February, 1991 Quarterly Traffic and }
                  { Capacity Data Master Data File Schedule P 05 }
  const
  num_ac = 76;
  eff_cst = 39.50;
  maxrec = 12000;

  type
   ac_car = (COA,NWA,UAL,AAL,USA,DAL,TWA,FDX,
             EAL,UPS,EIA,PCM,HAL,ASA,PAA,SWA,
             B1F,AMT,MID,AAH,AWI,QXE,SJM,APW,
             WOA,TPS,WCA,ASE,AWE,XXX);
             str1   = varying [1] of char;
             str3   = varying [3] of char;
             str4   = varying [4] of char;
             str5   = varying [5] of char;
             str7   = array [1..7] of char;
             str8   = array [1..8] of char;
             strr8  = varying [8] of char;
             str256 = varying [256] of char;
  var

     sum_aea, sum_aed, sum_dfx, sum_rst, sum_scc,
     sum_unacc_del_air, sum_afx, sum_apa, sum_apd, cost, sm_aea,
     sm_aed, sm_dfx, sm_rst, sm_scc, sm_afx, sm_apa, sm_apd : real;
     sum_ga_aea, sm_ga_aea, sum_ga_aed, sm_ga_aed,
     sum_ga_dfx, sm_ga_dfx, sum_ga_rst, sm_ga_rst,sum_unacc_del_ga,
     sum_ga_scc, sm_ga_scc, sum_ga_afx, sm_ga_afx,sum_unacc_del_mil,
     sum_ga_apa, sm_ga_apa, sum_ga_apd, sm_ga_apd,
     sum_mil_aea, sm_mil_aea, sum_mil_aed, sm_mil_aed,
     sum_mil_dfx, sm_mil_dfx, sum_mil_rst, sm_mil_rst,
     sum_mil_scc, sm_mil_scc, sum_mil_afx, sm_mil_afx,
     sum_mil_apa, sm_mil_apa, sum_mil_apd, sm_mil_apd     : real;
     error                                              : integer32;
     n1,n2,n3                                             : text;
     inf,fout,inf1,tout                                   : text;
     str36                                 : varying [36] of char;
     eq_typ_cd                           : array[1..90000] of str7;
     air_occ                               : array[1..75] of real;
     nmatchsort                          : array[1..maxrec] of str8;
     eqclass_ta                                         : integer;
     eqclass_er                                         : integer;
     k,i,n,l,code                                       : integer;
     tail_num                                           : integer32;
     tal_num                                            : str5;

     carrier,accar                                      : str3;
     actyp                                              : str4;
     fp_id                                              : integer;
     dly_typ                               : varying [3] of char;
```

```
delaystr                                                    : str7;
idx_typ,kkk                                               : integer;
delay                                                      : real;
sum_typ_del                          : array[1..num_ac] of real;
sum_car_del                             : array[ac_car] of real;
air_dly                    : array[ac_car,1..num_ac] of real;
grd_dly                    : array[ac_car,1..num_ac] of real;
acarr                                   : array[ac_car] of real;
atype                                : array[1..num_ac] of real;
Ind_carrier     : array  [COA..XXX] of varying [3] of char;
Ind_typ          : array[1..num_ac] of varying [4] of char;
jcar                                                       : ac_car;
jtyp                                                       : integer;
xxx,xjx,xx                                                 : integer;
absolutepath                                               : str256;
nomatch_file                                               : str256;
holdextension                                              : str256;
hold_name                                                  : strr8;
array_cost_name                                            : str256;
array_eq_cat_name                                          : str256;
trace_file_name                                            : str256;
results_file_name                                          : str256;


{ packs airborne delay costs into array air_dly }

procedure read_air_grd_cst;
var
carr : ac_car;
typp : integer;
carrra :  array[1..3] of char;
dummya : array[1..2] of char;
typppa : array[1..4] of char;
carrrg : array [1..3] of char;
dummyg : array [1..4] of char;
typppg : array [1..4] of char;

begin
writeln('Running Cost Module..... ',error);
argv( 1,array_cost_name);
reset( n2,array_cost_name);
for carr := COA to XXX do
begin
 for typp := 1 to num_ac do
  begin
   read(n2,carrra);
   read(n2,dummya);
   read(n2,typppa);
   readln(n2,air_dly[carr,typp]);
   end;
 end;
for carr := COA to XXX do
begin
 for typp := 1 to num_ac do
  begin
   read(n2,carrrg);
   read(n2,dummyg);
```

```
        read(n2,typpgg);
        readln(n2,grd_dly[carr,typp]);
        end;
       end;
      close(n2);
      end;


{fnd_typ converts tail number into carrier and aircraft type }
{if no match is found carrier and type are written to file nomatch}


procedure fnd_typ(var idx_typ : integer;var tail_num : integer32;
    var accar : str3);

    var
    atype  : varying[4] of char;
    kl     : integer;
    blank  : str1;
    begin
    if(tail_num < 90000) then
    begin
    blank := ' ';
    idx_typ := num_ac;
    atype := substr(eq_typ_cd[tail_num],4,4);
    accar := substr(eq_typ_cd[tail_num],1,3);

        for kl := 1 to num_ac do
          begin
           if(atype = Ind_typ[kl]) then
           idx_typ := kl;
          end; { for loop }
  if((idx_typ = num_ac) and (atype <>  '    ') and (kkk < maxrec)) then

       { write nomatch aircraft to array nmatchsort }

       begin
       kkk := kkk + 1;
       nmatchsort[kkk] := atype + blank + accar;
       end;
       end
      else
      writeln('tail number out of bounds - ',tail_num);
     end; { procedure }


{ convert string to real value }

    procedure str_to_real(var dlaystr : str7;var dly : real);
    var
    kk,ii,ip,it  : integer;
    val1 : array[1..3] of real;
    val2 : array[1..7] of real;
       begin
        for ip := 1 to 3 do
         val1[ip] := 0;
         for it := 1 to 7 do
```

```
        val2[it] := 0;
      dly := 0.0;
      for kk := 1 to 3 do
       begin
      case dlaystr[kk] of
      '0' : val1[kk] := 0;
      '1' : val1[kk] := 1;
      '2' : val1[kk] := 2;
      '3' : val1[kk] := 3;
      '4' : val1[kk] := 4;
      '5' : val1[kk] := 5;
      '6' : val1[kk] := 6;
      '7' : val1[kk] := 7;
      '8' : val1[kk] := 8;
      '9' : val1[kk] := 9;
      ' ' : val1[kk] := 0
      end;
      end;
          for ii := 5 to 7 do
          begin
          case dlaystr[ii] of
          '0' : val2[ii] := 0;
          '1' : val2[ii] := 1;
          '2' : val2[ii] := 2;
          '3' : val2[ii] := 3;
          '4' : val2[ii] := 4;
          '5' : val2[ii] := 5;
          '6' : val2[ii] := 6;
          '7' : val2[ii] := 7;
          '8' : val2[ii] := 8;
          '9' : val2[ii] := 9;
          ' ' : val2[ii] := 0
          end;
         end;
       dly := val1[1]*100.0 + val1[2]*10.0 + val1[3] +
      ((val2[5]*100 + val2[6]*10 + val2[7])/1000.0);
     end;

{ convert string to integer value }

 procedure str_to_integer(var tailnum : str5;var tailnumm : integer32);
    var
    kk1    : integer;
    value1 : array[1..5] of integer32;
       begin
        for kk1 := 1 to 5 do
         value1[kk1] := 0;
       tailnumm := 0;
       for kk1 := 1 to 5 do
        begin
       case tailnum[kk1] of
       '0' : value1[kk1] := 0;
       '1' : value1[kk1] := 1;
       '2' : value1[kk1] := 2;
       '3' : value1[kk1] := 3;
       '4' : value1[kk1] := 4;
       '5' : value1[kk1] := 5;
```

```
        '6' : value1[kk1] := 6;
        '7' : value1[kk1] := 7;
        '8' : value1[kk1] := 8;
        '9' : value1[kk1] := 9;
        ' ' : value1[kk1] := 0
       end;
       end;
       tailnumm := value1[1]*10000 + value1[2]*1000 + value1[3]*100 +
       value1[4]*10 + value1[5];
        end;

{equipment file contains tail number, carrier, and ac type}
{read carrier and type into array eq_typ_cat indexed by tail number}

    procedure read_eq_typ_cat;
    begin
    argv( 2, array_eq_cat_name);
    reset( inf1,array_eq_cat_name);
      repeat
        begin
        read(inf1,tal_num);
        str_to_integer (tal_num,tail_num);
         read(inf1,carrier);
         readln(inf1,actyp);
         eq_typ_cd[tail_num] := carrier + actyp;
          end;
      until eof(inf1);
      close(inf1);
    end;

    procedure process_data;
{reads data from TRACE FILE OF NASPAC model and start processing }
    var
     idx_typ                    : integer;
     idx_carr                   : ac_car;

     begin    { initialize variables for summing }

     sum_aea := 0; sum_aed := 0; sum_dfx := 0; sum_rst := 0;
     sum_scc := 0; sum_afx := 0; sum_apa := 0; sum_apd :=0;
     sm_dfx := 0; sm_rst := 0; sm_scc := 0; sm_afx := 0;
     sm_apa := 0; sm_apd :=0; sm_aea := 0; sm_aed := 0;
     sum_ga_aea := 0; sm_ga_aea := 0; sum_ga_aed := 0;
     sm_ga_aed := 0; sum_ga_dfx := 0; sm_ga_dfx := 0;
     sum_ga_rst := 0; sm_ga_rst := 0; sum_ga_scc := 0;
     sm_ga_scc := 0; sum_ga_afx := 0; sm_ga_afx := 0;
     sum_ga_apa := 0; sm_ga_apa := 0; sum_ga_apd := 0;
     sm_ga_apd := 0;
     sum_mil_aea := 0; sm_mil_aea := 0; sum_mil_aed := 0;
     sm_mil_aed := 0; sum_mil_dfx := 0; sm_mil_dfx := 0;
     sum_mil_rst := 0; sm_mil_rst := 0; sum_mil_scc := 0;
     sm_mil_scc := 0; sum_mil_afx := 0; sm_mil_afx := 0;
     sum_mil_apa := 0; sm_mil_apa := 0; sum_mil_apd := 0;
    sm_mil_apd := 0; sum_unacc_del_air := 0; sum_unacc_del_ga := 0;
    sum_unacc_del_mil := 0;

     while not eof(inf) do
```

```
begin
delay := 0; cost := 0;
readln(inf,str36);

   { read in trace file from scenario }
if (str36[5] = '.') then
  begin
   dly_typ  := substr (str36,9,3);
   tal_num := substr (str36,20,5);
   delaystr := substr (str36,30,7);
   str_to_real (delaystr,delay);
   str_to_integer (tal_num,tail_num);

   { convert tail number to aircraft type }

   fnd_typ(idx_typ,tail_num,carrier);

   { convert string to index of enumerated type }

   idx_carr := XXX;
   for jcar := COA to XXX do
   if (Ind_carrier[jcar] = carrier) then idx_carr := jcar;

   { sum delay and associated costs with delay type }
   { first see if there is a match }

   if(idx_typ = num_ac) then
   begin
   if(carrier = 'ga ') then sum_unacc_del_ga :=
     sum_unacc_del_ga + delay
    else
     if(carrier = 'ml ') then sum_unacc_del_mil :=
      sum_unacc_del_mil + delay
      else
      sum_unacc_del_air := sum_unacc_del_air + delay
      end;

   if(idx_typ < num_ac) then
   begin
   if(dly_typ = 'AEA' ) then

   { Airport Arrival Delay - " Effective Delay " }

   begin
   cost := eff_cst / 60.0 * delay * air_occ[idx_typ];
   if(carrier = 'ga ') then
    begin
    sum_ga_aea := sum_ga_aea + delay;
    sm_ga_aea := sm_ga_aea + cost;
        atype[idx_typ] := atype[idx_typ] + cost;
        sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
    end
      else
      if(carrier = 'ml ') then
       begin
       sum_mil_aea := sum_mil_aea + delay;
       sm_mil_aea := sm_mil_aea + cost;
```

```
      atype[idx_typ] := atype[idx_typ] + cost;
      sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
     end
      else
      begin
      sum_aea := sum_aea + delay;
      sm_aea := sm_aea + cost;
      acarr[idx_carr] := acarr[idx_carr] + cost;
      atype[idx_typ] := atype[idx_typ] + cost;
   sum_car_del[idx_carr] := sum_car_del[idx_carr] + delay;
   sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
      end
   end

else
if(dly_typ = 'AED') then

{ Airport Departure - " Effective Delay " }

 begin
  cost := eff_cst / 60.0 * delay * air_occ[idx_typ];
  if(carrier = 'ga ') then
 begin
 sum_ga_aed := sum_ga_aed + delay;
 sm_ga_aed := sm_ga_aed + cost;
     atype[idx_typ] := atype[idx_typ] + cost;
     sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
 end
   else
   if(carrier = 'ml ') then
    begin
    sum_mil_aed := sum_mil_aed + delay;
    sm_mil_aed := sm_mil_aed + cost;
     atype[idx_typ] := atype[idx_typ] + cost;
     sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
    end
     else
     begin
     sum_aed := sum_aed + delay;
     sm_aed := sm_aed + cost;
     acarr[idx_carr] := acarr[idx_carr] + cost;
     atype[idx_typ] := atype[idx_typ] + cost;
  sum_car_del[idx_carr] := sum_car_del[idx_carr] + delay;
  sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
    end
   end
   else

    if(dly_typ = 'DFX' ) then

      { Departure Fix Delay - " Airborne Delay "}

       begin
   { note if 0 cost is found for carrier, type go to type}
         if(air_dly[idx_carr,idx_typ] > 0.10) then
         cost := air_dly[idx_carr,idx_typ] * delay
         else
```

```
            cost := air_dly[XXX,idx_typ] * delay;
            if(carrier = 'ga ') then
            begin
            sum_ga_dfx := sum_ga_dfx + delay;
            sm_ga_dfx := sm_ga_dfx + cost;
            atype[idx_typ] := atype[idx_typ] + cost;
          sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
            end
     else
     if(carrier = 'ml ') then
      begin
      sum_mil_dfx := sum_mil_dfx + delay;
      sm_mil_dfx := sm_mil_dfx + cost;
           atype[idx_typ] := atype[idx_typ] + cost;
        sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
      end
       else
           begin
           sum_dfx := sum_dfx + delay;
           sm_dfx := sm_dfx + cost;
           acarr[idx_carr] := acarr[idx_carr] + cost;
           atype[idx_typ] := atype[idx_typ] + cost;
      sum_car_del[idx_carr] := sum_car_del[idx_carr] + delay;
      sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
           end
       end
        else
          if(dly_typ = 'RST' ) then

           { Restriction Delay - " Airborne Delay " }

            begin
        if(air_dly[idx_carr,idx_typ] > 0.10) then
        cost := air_dly[idx_carr,idx_typ] * delay
        else
        cost := air_dly[XXX,idx_typ] * delay;
        if(carrier = 'ga ') then
begin
sum_ga_rst := sum_ga_rst + delay;
sm_ga_rst := sm_ga_rst + cost;
     atype[idx_typ] := atype[idx_typ] + cost;
     sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
end
  else
  if(carrier = 'ml ') then
   begin
   sum_mil_rst := sum_mil_rst + delay;
   sm_mil_rst := sm_mil_rst + cost;
     atype[idx_typ] := atype[idx_typ] + cost;
     sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
   end
    else
     begin
     sum_rst := sum_rst + delay;
     sm_rst := sm_rst + cost;
     acarr[idx_carr] := acarr[idx_carr] + cost;
     atype[idx_typ] := atype[idx_typ] + cost;
```

```
sum_car_del[idx_carr] := sum_car_del[idx_carr] + delay;
    sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
    end
  end
     else

       if(dly_typ = 'SCC' ) then

          { Sector Entry Delay - " Airborne Delay " }

          begin
       if(air_dly[idx_carr,idx_typ] > 0.10) then
       cost := air_dly[idx_carr,idx_typ] * delay
       else
       cost := air_dly[XXX,idx_typ] * delay;
       if(carrier = 'ga ') then
begin
sum_ga_scc := sum_ga_scc + delay;
sm_ga_scc := sm_ga_scc + cost;
       atype[idx_typ] := atype[idx_typ] + cost;
  sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
end
  else
  if(carrier = 'ml ') then
   begin
   sum_mil_scc := sum_mil_scc + delay;
   sm_mil_scc := sm_mil_scc + cost;
       atype[idx_typ] := atype[idx_typ] + cost;
 sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
   end
                    else
       begin
       sum_scc := sum_scc + delay;
       sm_scc := sm_scc + cost;
       acarr[idx_carr] := acarr[idx_carr] + cost;
       atype[idx_typ] := atype[idx_typ] + cost;
  sum_car_del[idx_carr] := sum_car_del[idx_carr] + delay;
  sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
       end
     end
        else

            if(dly_typ = 'AFX' ) then

         { Arrival Fix Delay - " Airborne Delay " }

             begin
          if(air_dly[idx_carr,idx_typ] > 0.10) then
          cost := air_dly[idx_carr,idx_typ] * delay
          else
          cost := air_dly[XXX,idx_typ] * delay;
          if(carrier = 'ga ') then
begin
sum_ga_afx := sum_ga_afx + delay;
sm_ga_afx := sm_ga_afx + cost;
    atype[idx_typ] := atype[idx_typ] + cost;
    sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
```

```
end
  else
  if(carrier = 'ml ') then
   begin
   sum_mil_afx := sum_mil_afx + delay;
   sm_mil_afx := sm_mil_afx + cost;
            atype[idx_typ] := atype[idx_typ] + cost;
    sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
   end
    else
            begin
            sum_afx := sum_afx + delay;
            sm_afx := sm_afx + cost;
            acarr[idx_carr] := acarr[idx_carr] + cost;
            atype[idx_typ] := atype[idx_typ] + cost;
   sum_car_del[idx_carr] := sum_car_del[idx_carr] + delay;
   sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
            end
       end
             else

                if(dly_typ = 'APA' ) then

       { Airport Arrival Delay - " Airborne Delay " }

                   begin
               if(air_dly[idx_carr,idx_typ] > 0.10) then
               cost := air_dly[idx_carr,idx_typ] * delay
               else
               cost := air_dly[XXX,idx_typ] * delay;
               if(carrier = 'ga ') then
begin
sum_ga_apa := sum_ga_apa + delay;
sm_ga_apa := sm_ga_apa + cost;
    atype[idx_typ] := atype[idx_typ] + cost;
    sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
end
  else
  if(carrier = 'ml ') then
   begin
   sum_mil_apa := sum_mil_apa + delay;
   sm_mil_apa := sm_mil_apa + cost;
  atype[idx_typ] := atype[idx_typ] + cost;
 sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
   end
    else
               begin
               sum_apa := sum_apa + delay;
               sm_apa := sm_apa + cost;
               acarr[idx_carr] := acarr[idx_carr] + cost;
               atype[idx_typ] := atype[idx_typ] + cost;
  sum_car_del[idx_carr] := sum_car_del[idx_carr] + delay;
  sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
               end
          end
               else
```

```
                        if(dly_typ = 'APD' ) then

                { Airport Departure Delay - " Ground Delay " }

                            begin
                        if(grd_dly[idx_carr,idx_typ] > 0.10) then
                        cost := grd_dly[idx_carr,idx_typ] * delay
                        else
                        cost := grd_dly[XXX,idx_typ] * delay;
                        if(carrier = 'ga ') then
                        begin
                        sum_ga_apd := sum_ga_apd + delay;
                        sm_ga_apd := sm_ga_apd + cost;
                        atype[idx_typ] := atype[idx_typ] + cost;
          sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
                        end
                     else
                     if(carrier = 'ml ') then
                     begin
                     sum_mil_apd := sum_mil_apd + delay;
                     sm_mil_apd := sm_mil_apd + cost;
                        atype[idx_typ] := atype[idx_typ] + cost;
           sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay
                    end
                     else
                        begin
                        sum_apd := sum_apd + delay;
                        sm_apd := sm_apd + cost;
                        acarr[idx_carr] := acarr[idx_carr] + cost;
                    sum_car_del[idx_carr] := sum_car_del[idx_carr] + delay;
                    sum_typ_del[idx_typ] := sum_typ_del[idx_typ] + delay;
                        atype[idx_typ] := atype[idx_typ] + cost
                        end
                      end
                 else

                    end;
            end {if ( . )}
              end; {while not eof(inf)}

     { data processing over print results to a file fout }

 writeln(fout,'         Cost of Delay from NASPAC Simulation Results
Scenario ',hold_name);
 writeln(fout,' ');
 writeln(fout,' ');

     writeln(fout,'******************************     Effective  Delay
****************************');
     writeln(fout,' ');
     writeln(fout,'                                          General Aviation
',sum_ga_aea:12:0,' min',' cost',' $',sm_ga_aea:12:0);

     writeln(fout,'Effective Arrival Delay       Military
',sum_mil_aea:12:0,' min',' cost',' $',sm_mil_aea:12:0);

     writeln(fout,'                                          Air Carrier
```

```
',sum_aea:12:0,' min',' cost',' $',sm_aea:12:0);

     writeln(fout,'                                        Totals
',sum_ga_aea+sum_mil_aea+sum_aea:12:0,' min',' cost',' $',
     sm_ga_aea+sm_mil_aea+sm_aea:12:0);

     writeln(fout,' ');
     writeln(fout,'                                        General Aviation
',sum_ga_aed:12:0,' min',' cost',' $',sm_ga_aed:12:0);

     writeln(fout,'Effective Depart Delay          Military
',sum_mil_aed:12:0,' min',' cost',' $',sm_mil_aed:12:0);

     writeln(fout,'                                         Air Carrier
',sum_aed:12:0,' min',' cost',' $',sm_aed:12:0);

     writeln(fout,'                                        Totals
',sum_ga_aed+sum_mil_aed+sum_aed:12:0,' min',
                  ' cost',' $',sm_ga_aed+sm_mil_aed+sm_aed:12:0);


     writeln(fout,' ');
     writeln(fout,'************************ Technical Delay - Airborne
**********************');
     writeln(fout,' ');

     writeln(fout,'                                        General Aviation
',sum_ga_dfx:12:0,' min',' cost',' $',sm_ga_dfx:12:0);

     writeln(fout,'Departure Fix Delay             Military
',sum_mil_dfx:12:0,' min',' cost',' $',sm_mil_dfx:12:0);

     writeln(fout,'                                         Air Carrier
',sum_dfx:12:0,' min',' cost',' $',sm_dfx:12:0);

     writeln(fout,'                                        Totals
',sum_ga_dfx+sum_mil_dfx+sum_dfx:12:0,' min',
                  ' cost',' $',sm_ga_dfx+sm_mil_dfx+sm_dfx:12:0);

     writeln(fout,' ');

     writeln(fout,'                                        General Aviation
',sum_ga_rst:12:0,' min',' cost',' $',sm_ga_rst:12:0);

     writeln(fout,'Restriction    Delay            Military
',sum_mil_rst:12:0,' min',' cost',' $',sm_mil_rst:12:0);

     writeln(fout,'                                         Air Carrier
',sum_rst:12:0,' min',' cost',' $',sm_rst:12:0);

     writeln(fout,'                                        Totals
',sum_ga_rst+sum_mil_rst+sum_rst:12:0,' min',
                  ' cost',' $',sm_ga_rst+sm_mil_rst+sm_rst:12:0);

     writeln(fout,' ');

     writeln(fout,'                                        General Aviation
```

```
',sum_ga_scc:12:0,' min',' cost',' $',sm_ga_scc:12:0);

    writeln(fout,'Sector Entry   Delay                Military
',sum_mil_scc:12:0,' min',' cost',' $',sm_mil_scc:12:0);

    writeln(fout,'                                    Air Carrier
',sum_scc:12:0,' min',' cost',' $',sm_scc:12:0);

    writeln(fout,'                                    Totals
',sum_ga_scc+sum_mil_scc+sum_scc:12:0,' min',
                ' cost',' $',sm_ga_scc+sm_mil_scc+sm_scc:12:0);

    writeln(fout,' ');

    writeln(fout,'                                    General Aviation
',sum_ga_afx:12:0,' min',' cost',' $',sm_ga_afx:12:0);

    writeln(fout,'Arrival Fix    Delay                Military
',sum_mil_afx:12:0,' min',' cost',' $',sm_mil_afx:12:0);

    writeln(fout,'                                    Air Carrier
',sum_afx:12:0,' min',' cost',' $',sm_afx:12:0);

    writeln(fout,'                                    Totals
',sum_ga_afx+sum_mil_afx+sum_afx:12:0,' min',
                ' cost',' $',sm_ga_afx+sm_mil_afx+sm_afx:12:0);

    writeln(fout,' ');

    writeln(fout,'                                    General Aviation
',sum_ga_apa:12:0,' min',' cost',' $',sm_ga_apa:12:0);

    writeln(fout,'Airport Arrival Delay              Military
',sum_mil_apa:12:0,' min',' cost',' $',sm_mil_apa:12:0);

    writeln(fout,'                                    Air Carrier
',sum_apa:12:0,' min',' cost',' $',sm_apa:12:0);

    writeln(fout,'                                    Totals
',sum_ga_apa+sum_mil_apa+sum_apa:12:0,' min',
                ' cost',' $',sm_ga_apa+sm_mil_apa+sm_apa:12:0);

    writeln(fout,' ');
    writeln(fout,'************************** Technical Delay - Ground
************************');
    writeln(fout,' ');

    writeln(fout,'                                    General Aviation
',sum_ga_apd:12:0,' min',' cost',' $',sm_ga_apd:12:0);

    writeln(fout,'Airport Departure Delay            Military
',sum_mil_apd:12:0,' min',' cost',' $',sm_mil_apd:12:0);

    writeln(fout,'                                    Air Carrier
',sum_apd:12:0,' min',' cost',' $',sm_apd:12:0);

    writeln(fout,'                                    Totals
```

```
',sum_ga_apd+sum_mil_apd+sum_apd:12:0,' min',
                  ' cost',' $',sm_ga_apd+sm_mil_apd+sm_apd:12:0);


     writeln(fout,' ');

     writeln(fout,'***************************** Total  Delay  Costs
**************************');
     writeln(fout,'  ');

     writeln(fout,'                                    General Aviation
',sum_ga_apd+sum_ga_apa+sum_ga_afx+
      sum_ga_scc+sum_ga_rst+sum_ga_dfx+sum_ga_aed+sum_ga_aea:12:0,'
min',' cost',' $',sm_ga_aea+sm_ga_aed+
     sm_ga_dfx+sm_ga_rst+sm_ga_scc+sm_ga_afx+sm_ga_apa+sm_ga_apd:12:0);

     writeln(fout,'Total Delay Costs                Military
',sum_mil_apd+sum_mil_apa+sum_mil_afx

+sum_mil_scc+sum_mil_rst+sum_mil_dfx+sum_mil_aed+sum_mil_aea:12:0,'
min',' cost',' $',sm_mil_aea+

sm_mil_aed+sm_mil_dfx+sm_mil_rst+sm_mil_scc+sm_mil_afx+sm_mil_apa+sm_
mil_apd:12:0);

     writeln(fout,'                                    Air Carrier
',sum_apd+sum_apa+sum_afx+sum_scc+
      sum_rst+sum_dfx+sum_aed+sum_aea:12:0,'    min','    cost','
$',sm_aea+sm_aed+sm_dfx+sm_rst+sm_scc+
     sm_afx+sm_apa+sm_apd:12:0);

     writeln(fout,'                                    Grand Total
',sum_ga_aea+sum_mil_aea+sum_aea+

sum_ga_aed+sum_mil_aed+sum_aed+sum_ga_dfx+sum_mil_dfx+sum_dfx+sum_ga_
rst+sum_mil_rst+sum_rst+

sum_ga_scc+sum_mil_scc+sum_scc+sum_ga_afx+sum_mil_afx+sum_afx+sum_ga_
apa+sum_mil_apa+sum_apa+
      sum_ga_apd+sum_mil_apd+sum_apd:12:0,'    min','    cost','
$',sm_ga_aea+sm_mil_aea+sm_aea+

sm_ga_aed+sm_mil_aed+sm_aed+sm_ga_dfx+sm_mil_dfx+sm_dfx+sm_ga_rst+sm_
mil_rst+sm_rst+

sm_ga_scc+sm_mil_scc+sm_scc+sm_ga_afx+sm_mil_afx+sm_afx+sm_ga_apa+sm_
mil_apa+sm_apa+
     sm_ga_apd+sm_mil_apd+sm_apd:12:0);

      writeln(fout,' ');
      writeln(fout,'********************* Cost by Airlines and Aircraft
Types ******************');
      writeln(fout,' ');
   for jcar := COA to XXX do
      writeln(fout,'       Airlines ',Ind_carrier[jcar],'
',sum_car_del[jcar]
:14:0,' min','  cost $',acarr[jcar]:14:0);
```

```
      writeln(fout,' ');
            for jtyp := 1 to num_ac do
      writeln(fout,'         Aircraft Type ',Ind_typ[jtyp],'
',sum_typ_del
[jtyp]:14:0,' min',' cost $',atype[jtyp]:14:0);
      writeln(fout,' ');


      writeln(fout,' ');
      writeln(fout,'************ Unaccountable Delay - Measures Delay
with no Cost Information ************');

if (sum_unacc_del_ga > 0.001) then
      begin
      writeln(fout,' ');
writeln(fout,'General  Aviation     ',sum_unacc_del_ga:12:0,'  min','
representing ',sum_unacc_del_ga/(
sum_ga_aea+sum_ga_aed+
sum_ga_dfx+sum_ga_rst+
sum_ga_scc+sum_ga_afx+
sum_ga_apa+sum_ga_apd+sum_unacc_del_ga)*100:6:0,'  %  of  the  Total
Delay');
     end;

if (sum_unacc_del_mil > 0.001) then
      begin
      writeln(fout,' ');

writeln(fout,'Military              ',sum_unacc_del_mil:12:0,' min','
representing ',sum_unacc_del_mil/(
sum_mil_aea+sum_unacc_del_mil+
sum_mil_aed+sum_mil_dfx+sum_mil_rst+
sum_mil_scc+sum_mil_afx+
sum_mil_apa+sum_mil_apd)*100:6:0,' % of the Total Delay');
      end;

if (sum_unacc_del_air > 0.001) then
      begin
      writeln(fout,' ');

writeln(fout,'Air Carrier           ',sum_unacc_del_air:12:0,' min','
representing ',sum_unacc_del_air/(
sum_aea+sum_unacc_del_air+
sum_aed+sum_dfx+
sum_rst+sum_scc+sum_afx+
sum_apa+sum_apd)*100:6:0,' % of the Total Delay');
      end;

   if ((sum_unacc_del_mil+sum_unacc_del_air+sum_unacc_del_ga) > 0.001)
then
      begin
      writeln(fout,' ');

writeln(fout,'Total   Delay
',sum_unacc_del_air+sum_unacc_del_ga+sum_unacc_del_mil:12:0,' min',
'  representing ',(sum_unacc_del_air+
sum_unacc_del_ga+sum_unacc_del_mil)/(sum_unacc_del_air+sum_unacc_del_
```

```
ga+
sum_unacc_del_mil+
sum_ga_aea+sum_mil_aea+sum_aea+sum_ga_aed+
sum_mil_aed+sum_aed+sum_ga_dfx+sum_mil_dfx+sum_dfx+sum_ga_rst+sum_mil
_rst+
sum_rst+sum_ga_scc+sum_mil_scc+sum_scc+sum_ga_afx+sum_mil_afx+sum_afx
+
sum_ga_apa+sum_mil_apa+sum_apa+sum_ga_apd+sum_mil_apd+sum_apd)*100:6:
0,' % of the Total Delay');
      writeln(fout,' ');
      end;

      close (fout)
      end;


procedure keyswap(var rr,ss : str8);

 var
  t  : str8;

  begin

   t := rr;
   rr := ss;
   ss := t

  end;


  { sorts nomatch array }

procedure dosort(low, high : integer);

  var
   i,j   : integer;
   pivot : str8;

   begin
   if low < high then
     begin
      i := low;
      j := high;
      pivot := nmatchsort[j];
        repeat
           while ((i < j) and (nmatchsort[i] <= pivot)) do i := i + 1;
           while ((j > i) and (nmatchsort[j] >= pivot)) do j := j - 1;
           if i < j then keyswap(nmatchsort[i],nmatchsort[j]);
         until i >= j;
         keyswap(nmatchsort[i],nmatchsort[high]);
          if (i - low < high - i) then
               begin
               dosort(low,i-1);
               dosort(i + 1,high)
               end
                    else
```

```
                    begin
                     dosort(i + 1, high);
                     dosort(low,i-1)
                     end
                   end
          end;

{ writes sorted array to file nomatch and frequeny of occurance }

    procedure writsort;
    var
    knt,cnt    : integer;
    begin
    writeln(tout,'Missing Cost Information from Scenario ',hold_name);
    writeln(tout,'    ');
    writeln(tout,'Type Carrier Frequency');

     for knt := 1 to maxrec do
     begin
     if ( nmatchsort[knt] = nmatchsort[knt + 1]) then
     cnt := cnt + 1
      else
      begin
       if (nmatchsort[knt] <> '          ') then
       writeln(tout,' ',nmatchsort[knt],'   ',cnt :5);
      cnt := 1;
      end;
     end;
    close(tout);
    end;



       begin  {main}
    argv( 3,trace_file_name);
    reset( inf,trace_file_name);
    argv( 4,results_file_name);
    rewrite(fout,results_file_name);
    xxx := 1;
    while ( results_file_name[xxx] <> '.') do
       begin
       absolutepath := absolutepath + results_file_name[xxx];
       xxx := xxx + 1
       end;
    for xjx := xxx to 256 do
       begin
       holdextension := holdextension + results_file_name[xjx]
       end;
     for xjx := (xxx + 1) to (xxx + 8) do
       begin
          hold_name := hold_name + results_file_name[xjx]
       end;
      nomatch_file := absolutepath + '.' + '_nomatch' + holdextension
;
      open(tout,nomatch_file,'new');
      rewrite(tout);
      { initialize carrier and type arrays }
```

```
        for jcar := COA to XXX do
          begin
          Ind_carrier[jcar] := '   ';
          acarr[jcar] := 0;
          sum_car_del[jcar] := 0
          end;
        for jtyp := 1 to num_ac do
            begin
            Ind_typ[jtyp] := '    ';
            atype[jtyp] := 0;
            sum_typ_del[jtyp] := 0
            end;
         kkk := 0;
         for k := 1 to maxrec do
         nmatchsort[k] := '        ';

         for k := 1 to 90000 do
          eq_typ_cd[k] := '       ';
          { load carrier names }
     Ind_carrier[COA]:=    'COA';    Ind_carrier[AAL]:=    'AAL';
Ind_carrier[DAL]:= 'DAL';
     Ind_carrier[UAL]:=    'UAL';    Ind_carrier[NWA]:=    'NWA';
Ind_carrier[USA]:= 'USA';
     Ind_carrier[PAA]:=    'PAA';    Ind_carrier[EAL]:=    'EAL';
Ind_carrier[TWA]:= 'TWA';
     Ind_carrier[HAL]:=    'HAL';    Ind_carrier[B1F]:=    'B1F';
Ind_carrier[FDX]:= 'FDX';
     Ind_carrier[UPS]:=    'UPS';    Ind_carrier[EIA]:=    'EIA';
Ind_carrier[PCM]:= 'PCM';
     Ind_carrier[ASA]:=    'ASA';    Ind_carrier[SWA]:=    'SWA';
Ind_carrier[AMT]:= 'AMT';
     Ind_carrier[MID]:=    'MID';    Ind_carrier[AAH]:=    'AAH';
Ind_carrier[AWI]:= 'AWI';
     Ind_carrier[QXE]:=    'QXE';    Ind_carrier[SJM]:=    'SJM';
Ind_carrier[APW]:= 'APW';
     Ind_carrier[WOA]:=    'WOA';    Ind_carrier[TPS]:=    'TPS';
Ind_carrier[WCA]:= 'WCA';
     Ind_carrier[ASE]:=    'ASE';    Ind_carrier[AWE]:=    'AWE';
Ind_carrier[XXX]:= 'XXX';

     { load aircraft type names }

     Ind_typ[1]  :='DC9 '; Ind_typ[2]  :='B72S'; Ind_typ[3]  :='B737';
     Ind_typ[4]  :='SW4 '; Ind_typ[5]  :='BA14'; Ind_typ[6]  :='DC10';
     Ind_typ[7]  :='B747'; Ind_typ[8]  :='C310'; Ind_typ[9]  :='PA60';
     Ind_typ[10] :='B767'; Ind_typ[11] :='BE55'; Ind_typ[12] :='DH8 ';
     Ind_typ[13] :='B757'; Ind_typ[14] :='B727'; Ind_typ[15] :='BE1 ';
     Ind_typ[16] :='BE99'; Ind_typ[17] :='EM2 '; Ind_typ[18] :='DH6 ';
     Ind_typ[19] :='L101'; Ind_typ[20] :='E110'; Ind_typ[21] :='DC86';
     Ind_typ[22] :='A300'; Ind_typ[23] :='ATR '; Ind_typ[24] :='BA46';
     Ind_typ[25] :='F-28'; Ind_typ[26] :='F28 ';
     Ind_typ[27] :='DC7 '; Ind_typ[28] :='F27 '; Ind_typ[29] :='BA46';
     Ind_typ[30] :='SH7 '; Ind_typ[31] :='FA27'; Ind_typ[32] :='CV58';
     Ind_typ[33] :='DO28'; Ind_typ[34] :='SHD3'; Ind_typ[35] :='B707';
     Ind_typ[36] :='310 '; Ind_typ[37] :='BA11'; Ind_typ[38] :='FFJ ';
     Ind_typ[39] :='HS25'; Ind_typ[40] :='LR24'; Ind_typ[41] :='FA28';
     Ind_typ[42] :='YS11'; Ind_typ[43] :='100 '; Ind_typ[44] :='B74S';
```

```
Ind_typ[45] :='DC6 '; Ind_typ[46] :='AC2A'; Ind_typ[47] :='L188';
Ind_typ[48] :='DC8 '; Ind_typ[49] :='G73 '; Ind_typ[50] :='DH3 ';
Ind_typ[51] :='C550'; Ind_typ[52] :='BE20'; Ind_typ[53] :='BE90';
Ind_typ[54] :='PA31'; Ind_typ[55] :='BE58'; Ind_typ[56] :='LR35';
Ind_typ[57] :='PA28'; Ind_typ[58] :='BE36'; Ind_typ[59] :='BE30';
Ind_typ[60] :='BE10'; Ind_typ[61] :='PA32'; Ind_typ[62] :='C172';
Ind_typ[63] :='BE18'; Ind_typ[64] :='LR55'; Ind_typ[65] :='LR25';
Ind_typ[66] :='LR23'; Ind_typ[67] :='C421'; Ind_typ[68] :='C210';
Ind_typ[69] :='C650'; Ind_typ[70] :='C414'; Ind_typ[71] :='C182';
Ind_typ[72] :='C340'; Ind_typ[73] :='C500'; Ind_typ[74] :='C441';
Ind_typ[75] :='BE35'; Ind_typ[76] :='XXXX';

{ load occupancy figures for load factors }

air_occ[1] := 62; air_occ[2] := 91; air_occ[3] := 77; air_occ[4]
:= 62;
air_occ[5] := 51; air_occ[6] := 189; air_occ[7] := 234; air_occ[8]
:= 2;
air_occ[9] := 2; air_occ[10] := 129; air_occ[11] := 2; air_occ[12]
:= 11;
air_occ[13] := 117; air_occ[14] := 91; air_occ[15] := 3;
air_occ[16] := 2;
air_occ[17] := 3; air_occ[18] := 11; air_occ[19] := 177;
air_occ[20] := 3;
air_occ[21] := 135; air_occ[22] := 168; air_occ[23] := 125;
air_occ[24] := 51;
air_occ[25] := 2; air_occ[26] := 2; air_occ[27] := 125;
air_occ[28] := 2;
air_occ[29] := 51; air_occ[30] := 24; air_occ[31] := 2;
air_occ[32] := 3;
air_occ[33] := 3; air_occ[34] := 2; air_occ[35] := 54; air_occ[36]
:= 150;
air_occ[37] := 45; air_occ[38] := 3; air_occ[39] := 2; air_occ[40]
:= 3;
air_occ[41] := 2; air_occ[42] := 2; air_occ[43] := 3; air_occ[44]
:= 234;
air_occ[45] := 125; air_occ[46] := 3; air_occ[47] := 130;
air_occ[48] := 135;
air_occ[49] := 3; air_occ[50] := 3; air_occ[51] := 3; air_occ[52]
:= 2;
air_occ[53] := 3; air_occ[54] := 4; air_occ[55] := 3; air_occ[56]
:= 3;
air_occ[57] := 2; air_occ[58] := 3; air_occ[59] := 2; air_occ[60]
:= 3;
air_occ[61] := 3; air_occ[62] := 2; air_occ[63] := 2; air_occ[64]
:= 3;
air_occ[65] := 3; air_occ[66] := 3; air_occ[67] := 3; air_occ[68]
:= 2;
air_occ[69] := 3; air_occ[70] := 3; air_occ[71] := 2; air_occ[72]
:= 2;
air_occ[73] := 3; air_occ[74] := 4; air_occ[75] := 2;
```

```
read_air_grd_cst;
read_eq_typ_cat;
process_data;
dosort(1,maxrec);
writsort;
writeln('Cost Module is finished!!');
 end.  {program}
```

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
DOT/FAA/CT-TN91/52		

4. Title and Subtitle	5. Report Date
Cost of Delay Module	November 1991
	6. Performing Organization Code: ACD-340
	8. Performing Organization Report No.: DOT/FAA/CT-TN91/52
7. Author(s): Douglas Baart, Joseph M. Richie, and Kimberly A. May	
9. Performing Organization Name and Address: Federal Aviation Administration, Technical Center, Atlantic City International Airport, NJ 08405	10. Work Unit No. (TRAIS)
	11. Contract or Grant No.: F2006E
	13. Type of Report and Period Covered: Technical Note
12. Sponsoring Agency Name and Address: U.S. Department of Transportation, Federal Aviation Administration, Systems Analysis Division, Washington, DC 20591	14. Sponsoring Agency Code: ADR-100

15. Supplementary Notes

16. Abstract

This paper addresses the cost of the delay module that was developed by Federal Aviation Administration Technical Center to be incorporated into the National Airspace System Analysis Capability (NASPAC) model. This module was developed to address the savings which could be realized when changes are made to the Air Traffic Control (ATC) System. The purpose of this module is to translate delay into a cost metric and, thus, give policy makers a better understanding of potential cost saving measures. These cost saving measures are a direct result of an operational or procedural change to the ATC System. Cost estimates for major air carriers using the National Airspace System (NAS) were derived from Form 41 (operating expenses) data provided by the Office of Airline Statistics. General aviation and military cost estimates were derived from the Economic Values for Evaluation of Federal Aviation Administration Investment and Regulatory Programs, FAA-APO-89-10. The cost of the delay module has been tested and is fully operational with the latest release of NASPAC.

17. Key Words	18. Distribution Statement
Delay Module ATC NASPAC	Document is on file at the Technical Center Library, Atlantic City International Airport, NJ 08405

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	34	